UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/526,843	03/03/2005	Chenghui Luo	fraunh01.049	9357

7590 09/24/2008

Gordon E Nelson
Patent Attorney
57 Central Street
P O Box 782
Rowley, MA 01969

EXAMINER	
GYORFI, THOMAS A	

ART UNIT	PAPER NUMBER
2135	

MAIL DATE	DELIVERY MODE
09/24/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/526,843	LUO, CHENGHUI	
	Examiner	Art Unit	
	Thomas Gyorfi	2135	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *09 July 2008*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-29* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-29* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-29 remain for examination. The correspondence filed 7/9/08 amended claims 1-5 and 22-29.

Response to Arguments

2. Applicant's arguments filed 7/9/08 have been fully considered but they are not persuasive.

3. On page 13 of the amendment of 7/9/08, Applicant argues,

> Independent claims 1, 6, 14, and 22 all include: limitations which involve: detection of alterations in the software object or program being executed by the execution environment. Neither Monden nor Valdez contains any disclosure whatever concerning such techniques. Applicants use static and dynamic watermarks to detect such alterations; Monden adds static watermarks m his code, but as already pointed out in the *Response to the written opinion,* Monden's static watermark identifies the owner of the code. It contains no information about the code itself, and therefore cannot be used to detect alteration of the code. As for Valdez, Valdez gives a. taxonomy of "threats" at the bottom of page 382 and the top of page 383. One class of threats is described as follows:
>
> > Tampering, denoted as ALTER, is when unauthorized modification, such as adding, deleting, or changing instructions, of programs occurs.
>
> Two paragraphs further on, Valdez explicitly states that his paper does not address ALTER, i.e., discloses nothing about detecting alterations in the software objects or programs of Applicants' claims. Because neither reference discloses anything about detecting alterations on the software objects or programs of the claims: the combination of the references cannot disclose the claim limitations concerning detecting alterations and Examiner has not established his *prima facie* case with regard to the claims.

There are two problems with this argument:

(a) Ownership of Java programs is typically established by identifying the author of the program in the comments at the head of said program; the Java execution environment even provides a facility for automatically including such (see the enclosed JavaDoc reference, pages 8 & 11, and the sample code on pages 22-24). The Monden

disclosure would at the very least be able to identify that the authorship comments had been changed by a thief who would want to use the software object in question illegally (Monden, page 2, “Cases where watermark is needed”).

(b) Even if the claim language were written in such a way as to only cover non-trivial modifications that could otherwise materially affect the functionality of the software object (as appears to be argued by the Applicant), it is also noted that Java additionally provides the ability to watermark Java Applets (mobile code downloaded from a web site and executed on one’s web browser) with a digital signature to ensure that the actual software has not been tampered with (see the enclosed "A Functional Taxonomy for Software Watermarking" reference, page 179, "5.3.1 Validation Mark")

In essence, Applicant has attempted to distinguish the instant invention over the prior art by appealing to functionality inherent to the Java execution environment, and arguing that using said functionality in the manner for which it was intended would result in a novel and non-obvious invention. Examiner strongly disagrees, noting that if applying the techniques disclosed by Monden (in view of Valdez) on a Java program that already had authorship attribution and/or digitally signing an obfuscated Java program would lead to success, then the invention is not the product of innovation but of ordinary skill and common sense. *KSR v. Teleflex*, 550 U.S. at _, 82 USPQ2d at 1397.

4. On page 14 of the amendment of 7/9/08, Applicant argues,

> In combining Valdez with Monden in his rejection, Examiner admits that Monden discloses nothing whatever about the use of encryption to obfuscate symbolic names Valdez, however, also does not provide any disclosure concerning this limitation. As is clear from the paragraph at the middle of page 386 of Valdez, obfuscation and encryption are unrelated operations:
>
> > In a typical situation, when all the transformation primitives are selected, the hiding tool works as follows: First, obfuscating

> transformations then scrambling transformations are applied, Noise instructions are inserted either at random or at specified locations in the preselected sections .... Then the substitution primitive is applied. Original instructions are replaced with equivalent instructions .... After applying the noise and substitution primitives, the processed sections are decomposed into blocks of some finite size, and a random permutation order is applied on these blocks. Then from these blocks, blocks are randomly selected for encryption and/or compression ..... Note that these-transformation primitives can be interleaved and applied at different levels of granularity.

There is simply nothing here or anywhere else in Valdez that deals with the specific problem for which encryption is employed in Applicants' claims, namely effective obfuscation of symbolic names. Since that is the case, Examiner has also failed to make his *prima facie* case with regard to the claim limitations involving encryption that is employed to obfuscate symbolic names.

Examiner disagrees with Applicant's assessment, first by noting that the Applicant appears to have taken the cited passage out of context. Observe that section 5.1, "The Hiding Tool", first discloses:

> The core of the PHA is the preprocessing stage's hiding tool. This tool obfuscates and scrambles the content and runtime features of programs. Our basic hiding tool utilizes generic transformations that are generally applicable to all types of software executables (see Figure 2).
>
> The inputs to the basic hiding tool are the original program and a profile. The profile specifies the sections of the program to conceal and the concealment parameter. The concealment parameter specifies the type and degree of transformations to apply. For creating hidden programs, the basic hiding tool employs a straightforward procedure: a program is passed through a multistage process where on each stage a distinct transformation primitive is applied. The transformation primitives applied are noise, substitution, decomposition, permutation, compression, **and encryption**.

[emphasis Examiner's] In other words, encryption as employed by Valdez is done so precisely because it is recognized as a way to obfuscate code. Furthermore, observe that Valdez discloses that it is possible to operate the disclosed hiding tool by setting the type and degree of transformations to "encryption" and "100%" respectively (see Table 5 on page 391). When operated in this fashion, the disclosed hiding tool by definition encrypts every aspect of the program being obfuscated, which necessarily

includes the symbolic names of the variables used therein (and the example scripts being obfuscated all contain at least one variable, as seen in pages 390-391).

Claim Rejections - 35 USC § 103

5. The text of those sections of Title 35, U.S. Code not included in this action can be found in a prior Office action.

6. Claims 1-29 are rejected under 35 U.S.C. 103(a) as being unpatentable over “A Practical Method for Watermarking Java Programs” from the IDS filed 8/15/05 (hereinafter, “Monden”) in view of "Software DisEngineering: Program Hiding Architecture and Experiments" (hereinafter, "Valdez").

Regarding claim 1:

Monden discloses code comprising: one or more obfuscated names that correspond to system symbolic names (page 192, “3.1 Current solutions for program theft”, third paragraph); a first association between the obfuscated names and other forms of the corresponding symbolic names (page 194, Figures 4 & 5); a static watermark that has been added to the code (Ibid, and “Watermark injection”); the execution environment including a second association of the forms with information needed to resolve the corresponding system symbolic names, using the first and second associations to resolve the obfuscated names, and using the static watermark to determine the authenticity of the code (Ibid, and page 195, “Decoding Procedure”).

Monden does not explicitly disclose that any of the corresponding system symbolic names are encrypted. However, Valdez discloses a number of techniques for preventing reverse engineering of software for illegal uses (page 1, "Introduction"; cf. Monden, page 1, "Introduction") wherein information in a program, including symbolic information can be encrypted to obscure it (pages 386-387, "5.2 Execution Credential and Descrambling"; and Tables 1-5). It would have been obvious to encrypt symbolic information in a Java program as part of the watermarking process disclosed by Monden, because the encryption technique(s) disclosed by Valdez were part of the ordinary capabilities of one of ordinary skill in the art, in view of the teaching of the technique(s) for improvement in related situations.

Examiner takes Official Notice that any alleged discrepancies between the claim(s) and the prior art could easily be incorporated into the prior art combination via the use of functionality well known to be present in the Java execution environment (see the enclosed JavaDoc and Nagra references; cf. pages 2-3 above).

Regarding claims 6 and 14:

Monden teaches an improved class loader (class loaders in general being an inherent component to Java) and method comprising: using the first association and a second association between the forms and information used to resolve the symbolic names defined in the class to resolve the symbolic names in a program (pages 194-195, including Figures 4-8); and adding a method to the program which determines

whether the program has been modified by the host (page 194, “Dummy method injection”).

Monden does not explicitly disclose that any of the corresponding system symbolic names are encrypted. However, Valdez discloses a number of techniques for preventing reverse engineering of software for illegal uses (page 1, "Introduction”; cf. Monden, page 1, “Introduction”) wherein information in a program, including symbolic information can be encrypted to obscure it (pages 386-387, “5.2 Execution Credential and Descrambling”; and Tables 1-5). It would have been obvious to encrypt symbolic information in a Java program as part of the watermarking process disclosed by Monden, because the encryption technique(s) disclosed by Valdez were part of the ordinary capabilities of one of ordinary skill in the art, in view of the teaching of the technique(s) for improvement in related situations.

Examiner takes Official Notice that any alleged discrepancies between the claim(s) and the prior art could easily be incorporated into the prior art combination via the use of functionality well known to be present in the Java execution environment (see the enclosed JavaDoc and Nagra references; cf. pages 2-3 above).

Regarding claim 22:

Monden a method comprising: replacing symbolic names in the program that are defined in the class with obfuscated symbolic names corresponding thereto (page 194, “Watermark injection”, and Figures 4-5); making a first association between the obfuscated symbolic names and forms of the replaces symbolic names (Ibid); making a

second association between the forms of the symbolic names and information required to resolve the symbolic names (page 195, Figures 6-8); adding a method to the program that determines whether the program has been modified by the host (page 194, “Dummy method injection”); using the first and second associations to resolve the obfuscated symbolic names (page 195, “Decoding Procedure") and executing the added method to determine whether the program has been modified by the host (Ibid).

Monden does not explicitly disclose that any of the corresponding system symbolic names are encrypted. However, Valdez discloses a number of techniques for preventing reverse engineering of software for illegal uses (page 1, "Introduction”; cf. Monden, page 1, “Introduction”) wherein information in a program, including symbolic information can be encrypted to obscure it (pages 386-387, “5.2 Execution Credential and Descrambling”; and Tables 1-5). It would have been obvious to encrypt symbolic information in a Java program as part of the watermarking process disclosed by Monden, because the encryption technique(s) disclosed by Valdez were part of the ordinary capabilities of one of ordinary skill in the art, in view of the teaching of the technique(s) for improvement in related situations.

Examiner takes Official Notice that any alleged discrepancies between the claim(s) and the prior art could easily be incorporated into the prior art combination via the use of functionality well known to be present in the Java execution environment (see the enclosed JavaDoc and Nagra references; cf. pages 2-3 above).

Regarding claims 2 and 10:

Monden further discloses wherein the static watermark's value is a digest of the code prior to the addition of the static watermark (Figure 6).

Regarding claims 3 and 23:

Valdez further discloses wherein other obfuscated names that replace names defined in source code from which the code was made (page 381, "Transformations").

Regarding claim 4:

Monden further discloses wherein the code is downloaded to the program execution environment for execution (e.g. Java applets on Internet sites, page 191).

Regarding claims 5, 13, 21, and 29:

Valdez further discloses wherein the program includes an encrypted form of the encryption key used to produce the second association (the session key, page 380), and the improved class loader obtains the encryption key by using a decryption key to decrypt the encrypted form of the encryption key (decryption key, page 382).

Regarding claim 7:

Valdez further discloses wherein the method is encrypted prior to being added to the program and the improved class loader decrypts the method on adding the method to the program (page 388, "6.1 TCL Transformations", particularly 1st paragraph).

Regarding claim 8:

Monden further discloses wherein the program includes information from which the method determines whether the program has been modified by the host (page 194, "Dummy method injection")

Regarding claims 9, 17, and 26:

Monden further discloses wherein the program includes a static watermark (pages 194-195, "Watermark injection") and the static watermark is the information from which the method determines whether the program has been modified by the host (Ibid)

Regarding claims 11, 19, and 27:

Monden further discloses wherein the static watermark is at a location in the program determined by a key (Figures 6, 7, & 8) and the method has access to the key and uses the key to locate the static watermark (Ibid).

Regarding claims 12, 20, and 28:

Valdez further discloses wherein the improved class loader has access to an encryption key that was used to produce the encrypted forms in the first association (pages 386-387, "5.2 Execution Credential and Descrambling"); and the improved class loader uses encryption key to produce second association on loading the class (Ibid).

Regarding claims 15 and 24:

Valdez further discloses wherein the added method is encrypted (pages 386-387, "5.2 Execution Credential and Descrambling"; cf. Tables 1-5); and the step of adding the method includes the step of decrypting the method (Ibid).

Regarding claim 16:

Monden further discloses wherein the program includes information which the added method uses to determine whether the program has been modified by the host (the watermark: see page 195, including "4.2 Decoding Procedure").

Regarding claim 18:

Monden further discloses wherein the static watermark's value is a digest of the code prior to the addition of the static watermark (Figure 6) and wherein the added method reads the static watermark, recomputes the digest, and compares the recomputed digest with the watermark's value (page 195 "Decoding procedure").

Regarding claim 25:

Monden further discloses wherein the program includes information which the added method uses to determine whether the program has been modified by the host (the watermark: see page 195, including "4.2 Decoding Procedure") and in the step of executing the added method, the added method uses the information to determine whether the program has been modified by the host (Ibid).

Conclusion

7. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Thomas Gyorfi whose telephone number is (571)272-3849. The examiner can normally be reached on 8:30am - 5:00pm Monday - Friday.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kim Vu can be reached on (571) 272-3859. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

TAG
9/4/08

/KimYen Vu/

Supervisory Patent Examiner, Art Unit 2135